PATENT ABSTRACTS OF JAPAN

(11)Publication number : 07-311753
(43)Date of publication of application : 28.11.1995

(51)Int.Cl. G06F 15/177
G06F 9/06
G06F 9/22
G06F 13/00

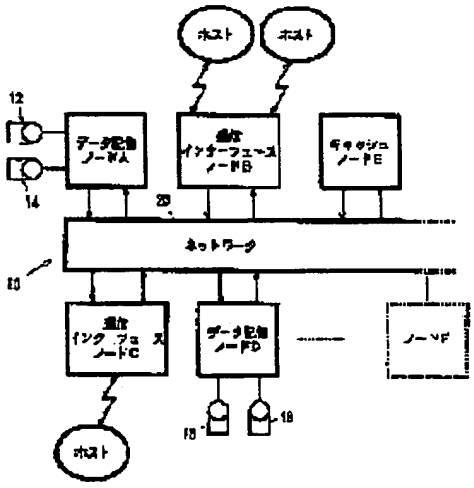
(21)Application number : 07-107474
(22)Date of filing : 01.05.1995

(71)Applicant : INTERNATL BUSINESS MACH CORP <IBM>
(72)Inventor : JOHN D EAGER
LAWRENCE Y HO
CHESTER R STEVENS
BRADY JAMES T
DAVID T WANG

(30)Priority
Priority number : 94 241901 Priority date : 11.05.1994 Priority country : US

(54) METHOD FOR UPDATING CONTROL CODE IN PLURAL NODES AND DEVICE THEREFOR

(57)Abstract:
PURPOSE: To attain code update enabling communication among nodes with code change in different levels in a multinode system.
CONSTITUTION: The update of a control code is executed in many nodes in a calculation system while the calculation system is operated. Each node includes a processor, memory, first version of a control code unit, and instruction of a technical change level related with the control code unit. This method includes a step for introducing the revised version of the control unit with a conversion program code module to a first node, and the conversion program code module validates and executes first and second interface functions during communication between the first node and the other nodes in the system.



LEGAL STATUS
[Date of request for examination] 29.10.1997
[Date of sending the examiner's decision of rejection]
[Kind of final disposal of application other than

(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号

特開平7－311753

(43)公開日　平成７年(1995)11月28日

(51)Int.Cl.[6]	識別記号	庁内整理番号	ＦＩ	技術表示箇所
Ｇ０６Ｆ　15/177				
9/06	５４０　Ｆ	7230－5B		
9/22	３７０	7230－5B		
13/00	３５５	7368－5B		
			Ｇ０６Ｆ　15/16　４２０　Ｓ	

審査請求　未請求　請求項の数12　ＯＬ　（全 11 頁）

(21)出願番号　特願平7－107474

(22)出願日　平成７年(1995)５月１日

(31)優先権主張番号　２４１９０１

(32)優先日　1994年５月11日

(33)優先権主張国　米国（ＵＳ）

(71)出願人　390009531
インターナショナル・ビジネス・マシーンズ・コーポレイション
ＩＮＴＥＲＮＡＴＩＯＮＡＬ　ＢＵＳＩＮＥＳＳ　ＭＡＳＣＨＩＮＥＳ　ＣＯＲＰＯＲＡＴＩＯＮ
アメリカ合衆国10504、ニューヨーク州　アーモンク　（番地なし）

(72)発明者　ジョン・ディー・イーガー
アメリカ合衆国95124　カリフォルニア州　サンノゼ　ジャスティン・ドライブ　3607

(74)代理人　弁理士　合田　潔　（外２名）

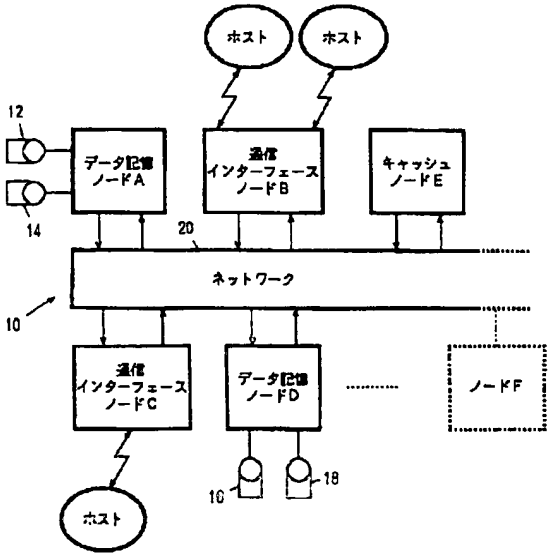
最終頁に続く

(54)【発明の名称】　複数のノード内の制御コードを更新する方法および装置

(57)【要約】

【目的】　コード変更が異なるレベルにあるノード間で通信を可能にするコード更新をマルチノード・システムにおいて可能にする。

【構成】　計算システムが動作したままで計算システムの多数のノード内で制御コードの更新が実施される。各ノードは、プロセッサ、メモリ、制御コード単位の第1のバージョン、および制御コード単位に関する技術変更レベルの指示を含む。この方法は、第1のノードに変換プログラム・コード・モジュールと共に制御コード単位の改訂バージョンを導入する段階を含み、変換プログラム・コード・モジュールは、第1のノードとシステム内の他のノードとの間の通信中に、第1と第2のインターフェース機能を使用可能にして実行する。



1

【特許請求の範囲】

【請求項1】多数のノードが通信ネットワークによって相互接続され、各ノードが、プロセッサ、メモリ、制御コード単位の第1のバージョン、および前記制御コード単位に関する技術変更レベル(ECL)値を含み、1つのノードが除去されたときでも動作可能な、計算システムが動作したままで計算システムにおける複数のノード内の制御コードを更新する方法であって、

a．前記計算システムから第1のノードを分離する段階と、

b．前記制御コード単位の改訂バージョンと、前記第1のノード内の前記プロセッサと協力して動作し前記第1のノードと第2のノードとの通信中に第1と第2のインターフェース機能を実行する変換プログラム・コード・モジュールとを、前記第1のノードのメモリに導入して操作する段階と、

c．前記第1のノードを前記計算システムに結合する段階と、

d．前記第1のノードを操作して前記第2のノードとの通信を必要とする機能を実行する段階とを含み、前記第1のノード内の前記変換プログラム・コード・モジュールが前記第2のノードに記憶されたECL値を決定し、前記第1と第2のノード内のECL値が一致する場合は、前記第1のインターフェース機能を使って前記第2のノードと通信し、前記第1と第2のノード内のECL値が一致しない場合は、前記第2のインターフェース機能を使って前記第2のノードと通信し、それにより、前記制御コード単位の更新中に前記計算システムを動作可能にする方法。

【請求項2】e．前記計算システムから前記第2のノードを分離する段階と、

f．前記制御コード単位の改訂バージョンと、前記第2のノード内の前記プロセッサと協力して動作し前記第2のノード、第1のノードおよびその他のノードの間の通信中に第1と第2のインターフェース機能を使用可能にし実行する変換プログラム・コード・モジュールとを、前記第2のノードのメモリに導入する段階と、

g．前記第2のノードを前記計算システムに結合する段階と、

h．前記第1と第2のノードを操作して、前記第1のノード、第2のノードおよびその他のノードの間の通信を必要とする機能を実行する段階とをさらに含み、前記変換プログラム・コード・モジュールが、前記第1と第2のノードに記憶されたECL値の一致を確認したときには、前記第1のインターフェース機能を使って前記第1と第2のノードの間の通信を可能し、前記その他のノードとのECL値の不一致を確認したときは、前記第2のインターフェース機能を使って前記その他のノードと通信し、それより、前記ノード全部の前記制御コード単位の更新中に前記第1、第2およびその他のノードを動作

2

可能にする、請求項1に記載の方法。

【請求項3】計算システムが、前記計算システムのノードに修正された前記制御コード単位を導入する順序を定義するシーケンサ・コード・モジュールを含み、

前記制御コード単位の前記改訂バージョンを、前記シーケンサ・コード・モジュールによって指定されたように前記ノードに導入する段階を含むことを特徴とする、請求項1に記載の方法。

【請求項4】あるノードが、前記制御コード単位の第1バージョンおよび改訂バージョンを有する他のノードとの相互通信を必要とする場合、そのノードに導入された変換プログラム・コード・モジュールが、インバウンドおよびアウトバウンドのコード変換プログラム・サブモジュールを含むことを特徴とする、請求項2に記載の方法。

【請求項5】複数のノードが通信ネットワークによって相互接続され、各ノードが、プロセッサ、メモリ、制御コード単位の第1のバージョン、および前記制御コード単位に関する技術変更レベル(ECL)値を含み、1つのノードが除去されたときでも動作可能な、計算システムが動作したままで計算システムにおける複数のノード内の制御コードを更新する装置であって、

前記第1のノードを前記計算システムから切り離した後、前記制御コード単位の改訂バージョンと、前記第1のノード内の前記プロセッサと協力して動作し前記第1のノードと第2のノードとの通信中に第1と第2のインターフェース機能を実行する変換プログラム・コード・モジュールとを、前記第1のノードのメモリに導入する手段と、

前記第1のノードを前記計算システムに再結合し、前記第1のノードを操作して、前記第2のノードとの通信を必要とする機能を実行する手段と、

前記第1のノード内の前記変換プログラム・コード・モジュールを操作して、前記第2のノードに記憶されたECL値を決定し、前記第1と第2のノード内のECL値が一致する場合は、前記第1のインターフェース機能を使って前記第2のノードと通信し、前記第1と第2のノード内のECL値が一致しない場合は、前記第2のインターフェース機能を使って前記第2のノードと通信する手段とを含む装置。

【請求項6】前記計算システムから前記第2のノードを切り離す際に、前記制御コード単位の改訂バージョンと、前記第2のノード内の前記プロセッサと協力して動作し前記第2のノード、第1のノードおよびその他のノードの間の通信中に第1と第2のインターフェース機能を使用可能にして実行する変換プログラム・コード・モジュールとを、前記第2のノードのメモリに導入する手段と、

前記第2のノードを前記計算システムに再結合した後で動作可能な、前記第1と第2のノードを操作して、前記

3

第1 のノード、第2 のノードおよびその他のノードの間の通信を必要とする機能を実行する手段と、
前記第1 と第2 のノードに一致するＥＣＬ値が記憶されているかどうか決定し、一致するＥＣＬ値が記憶されている場合は、前記制御コード・モジュール内の前記第1 のインターフェース機能を使って前記第1 と第2 のノードの間の通信を可能し、前記第2 のノードおよび前記その他のノード内に不一致のＥＣＬ値が確認された場合は、前記制御コード・モジュール内の前記第2 のインターフェース機能を使って前記その他のノードと通信する手段とをさらに含むことを特徴とする、請求項5 に記載の装置。
【請求項7 】前記装置が、修正された前記制御コード単位を前記計算システムのノード内に導入する順序を定義するシーケンサ・コード・モジュールを含み、
各ノードにおいて、前記制御コード単位の前記改訂バージョンを、前記シーケンサ・コード・モジュールによって指定された順序で導入するために前記シーケンサ・コード・モジュールと協力して動作する手段を含むことを特徴とする、請求項6 に記載の装置。
【請求項8 】あるノードが、前記制御コード単位の第1 バージョンおよび改訂バージョンを有する他のノードとの相互通信を必要とする場合、そのノードに導入された変換プログラム・コード・モジュールが、インバウンドおよびアウトバウンドのコード変換プログラム・サブモジュールを含むことを特徴とする請求項7 に記載の装置。
【請求項9 】複数のノードが通信ネットワークによって相互接続され、各ノードが、プロセッサ、メモリ、前記ノード内の動作マイクロコードのコード・モジュールである制御コード単位の第1 のバージョン、および前記制御コード単位に関する技術変更レベル（ＥＣＬ）値を含み、計算システムが動作したままで計算システムにおける多数のノード内の制御コードを更新する方法であって、
ａ．第1 のノード内の前記動作マイクロコードから制御コード単位を分離する段階と、
ｂ．前記分離した制御コード単位の代わりに、前記制御コード単位の改訂バージョンと、前記第1 のノード内の前記プロセッサと協力して動作し前記第1 のノードと第2 のノードとの通信中に第1 と第2 のインターフェース機能を実行する変換プログラム・コード・モジュールとを、前記第1 のノード内のメモリに導入し操作する段階と、
ｃ．前記第1 のノードを操作して前記第2 のノードとの通信を必要とする機能を実行する段階とを含み、前記第1 のノード内の前記変換プログラム・コード・モジュールが前記第2 のノードに記憶されたＥＣＬ値を決定し、前記第1 と第2 のノード内のＥＣＬ値が一致する場合は、前記第1 のインターフェース機能を使って前記第2

4

のノードと通信し、前記第1 と第2 のノード内のＥＣＬ値が一致しない場合は、前記第2 のインターフェース機能を使って前記第2 のノードと通信し、それにより、前記制御コード単位の更新中に前記計算システムを動作可能にする方法。
【請求項１０】ｄ．前記第2 のノード内の前記マイクロコードから前記制御コードを分離する段階と、
ｅ．前記分離された制御コード単位の代わりに、前記制御コード単位の改訂バージョンと、前記第2 のノード内の前記プロセッサと協力して動作し前記第2 のノード、第1 のノードおよびその他のノードの間の通信中に第1 と第2 のインターフェース機能を使用可能にし実行する変換プログラム・コード・モジュールとを、前記第2 のノード内のメモリに導入する段階と、
ｆ．前記修正された前記制御コード単位を前記第2 のノード内の動作マイクロコードに結合する段階と、
ｇ．前記第1 と第2 のノードを操作して、前記第1 のノード、第2 のノードおよびその他のノードの間での通信を必要とする機能を実行する段階とをさらに含み、前記変換プログラム・コード・モジュールが、前記第1 と第2 のノードに記憶されたＥＣＬ値の一致を確認したときには、前記第1 のインターフェース機能を使って前記第1 と第2 のノードの間の通信を可能し、前記その他のノードとのＥＣＬ値の不一致を確認したときは、前記第2 のインターフェース機能を使って前記その他のノードと通信し、それより、前記第1 、第2 およびその他のノードを、前記ノード全部の前記制御コード単位の更新中に動作可能にする、請求項9 に記載の方法。
【請求項１１】計算システムが、前記計算システムのノードに前記制御コード単位の改訂バージョンを導入する順序を定義するシーケンサ・コード・モジュールを含み、
前記制御コード単位の改訂バージョンを、前記シーケンサ・コード・モジュールによって指定されるように前記ノードに導入する段階を含むことを特徴とする、請求項9 に記載の方法。
【請求項１２】前記ノードが、前記制御コード単位の第1 バージョンおよび改訂バージョンを有する他のノードとの相互通信を必要とする場合、ノード内に導入された変換プログラム・コード・モジュールが、インバウンドおよびアウトバウンドのコード変換プログラム・サブモジュールを含むことを特徴とする、請求項9 に記載の方法。
【発明の詳細な説明】
【０００１】
【産業上の利用分野】本発明は、マルチノード・ネットワークに関し、より詳細には、修正されたマイクロコードがネットワークの1 つまたは複数のノードに導入される間、ネットワークが動作し続けることを可能にする装置および手順に関する。

5

【０００２】

【従来の技術】分散処理システムが複雑になるにつれて、システム使用可能度に関する顧客要求もまた厳しくなってきた。分散処理システムは多数のノード（たとえば、数百個から数千個程度）を含み、各ノードはプロセッサと様々な支援モジュールを含む。分散処理システムは、システムの効率的な動作を可能にするために、数メガバイトにも及ぶ制御コードを必要とすることがある。そのようなシステムに機能を追加すると、制御コードのサイズは数１０メガバイトにも増大することがある。そのような大きな制御コードは、変更、更新、修正などを常に必要とする。コードの変更が導入される度に分散処理システムの使用の中断が必要である場合、顧客のシステム利用は著しく妨げられる。

【０００３】従来技術は、システムの操作性をある程度維持しながら制御コードに対する更新の導入を可能にする様々な技法を述べている。リュウ（Ｌｉｕ）他による米国特許第５１５５８３７号明細書は、アプリケーション・プログラムまたはオペレーティング・システム・プログラムを使用を中断せずに更新できる時分割マルチ・プロセッサ・システムを記載している。システム内のプロセッサは、２つの論理区画に分割される。旧バージョンのソフトウェアが一方の区画で稼働し、同時に新バージョンが他方の区画にロードされて始動される。新バージョンが適切に動作していることが確認されると、旧バージョンの区画から新バージョンの区画にデータ・トラフィックが2段階で転送される。最初に、入力データが新バージョンに切り換えられる。旧バージョンで進行中であったトランザクションがすべて完了したとき、出力データが旧バージョンから新バージョンに切り換えられる。

【０００４】ビーバートン（Ｂｅａｖｅｒｔｏｎ）他による米国特許第５２１０８５４号明細書は、プログラム可能読取り専用メモリに記憶されたプログラムを更新するシステムを記載している。更新手順の間、サブルーチンの新バージョンが、プログラム可能読取り専用メモリの空き区域に記憶される。この記憶は、システム・ファームウェアの保護された区画への書込みを防ぐため、制御装置がプログラム可能読取り専用メモリに常駐するファームウェアを区分した後で行われる。転送ベクトルを使って、ファームウェアに常駐するサブルーチンの間接アドレス指定が行われれる。サブルーチンの更新バージョンが記憶された後、サブルーチンの旧バージョンを指示する転送ベクトルが、新バージョンを示すように更新される。

【０００５】要するに、従来の処理システムにおけるマイクロコードの変更は、一般に計算機の運転停止を必要とし、その結果、顧客の混乱を招いた。最近の製品においては、ノードまたは計算機の2つの同一のクラスタがある場合、マイクロコードの変更は、各クラスタで一時

6

に1つずつ活動化される。システムの一方の半分が更新される間、他の半分は同時に独立して動作する。しかし、システムの2つの部分の間の通信は、2つの部分が異なる変更レベルにあるときは切断される。

【０００６】

【発明が解決しようとする課題】したがって、本発明の目的は、コード変更が異なるレベルにあるノード間で通信を可能にするコード更新をマルチノード・システムにおいて可能にすることである。

【０００７】本発明のもう１つの目的は、導入処理の間もマルチノード・システムが動作し続けるように、制御コードの更新を導入する、マルチノード・システム用の装置を提供することである。

【０００８】本発明のもう１つの目的は、コードの改訂が所定の順序に従って導入される、マルチノード・システムにコードの改訂を導入するための方法および装置を提供することである。

【０００９】

【課題を解決するための手段】計算システムが動作したままで計算システムの多数のノード内で制御コードの更新が実施される。各ノードは、プロセッサ、メモリ、制御コード単位の第1のバージョン、および制御コード単位に関する技術変更レベルの指示を含む。この方法は、第1のノードに変換プログラム・コード・モジュールと共に制御コード単位の改訂バージョンを導入する段階を含み、変換プログラム・コード・モジュールは、第1のノードとシステム内の他のノードとの間の通信中に、第1と第2のインターフェース機能を使用可能にして実行する。次に、第1のノードが、他のノードとの通信に必要とされる機能を実行するように操作され、第1のノード内の変換プログラム・コード・モジュールは、まず、別のノードに記憶された技術変更レベル値を決定し、ノード内の技術変更レベル値が一致する場合は、第1のインターフェース機能を使って他のノードと通信する。技術変更レベル値が一致しないことを確認した場合は、第2のインターフェース機能を使って他のノードとの通信を行い、異なるレベルのコード変更がある場合でも、両方のノードが通信できるようにする。また、変換プログラム・コード・モジュールに課される要件を簡略化するために、指定された順序ですべてのノードの更新を可能にするシーケンサが計算システムに設けられる。

【００１０】

【実施例】図1は、マルチノード・ネットワークとして構成されたディスク・ドライブ・アレイ１０を示す。ノードＡとＤは、対になったディスク・ドライブ１２と１４、および１６と１８にそれぞれ接続するデータ記憶ノードである。ディスク・ドライブは4台だけ示したが、ディスク・ドライブ・アレイ１０がさらに多数のディスク・ドライブ（およびさらに多くのノード）を含むことは当業者には理解されよう。１対の通信インターフェー

ス・ノードＢおよびＣが、アレイに入出力通信機能を提供する。ホスト・プロセッサが、通信リンクを介してノードＢとＣに結合されている。さらにアレイ１０は、ディスクとの入力出力両方のデータ転送に一時記憶機能を提供するキャッシュ・ノードＥを含む。アレイ１０は、他のノード（たとえば、ノードＦ）を追加することによって拡張でき、それらのノードはすべて通信ネットワーク２０によって相互接続される。

【００１１】ノードＡ～Ｆはそれぞれ、図2に示した通常のノード記憶で構成される。ノードは、ノードの全体的動作を制御するマイクロプロセッサ２２を含む。メモリ・インターフェース・モジュール２４は、マイクロプロセッサ２２とノード内の複数のメモリ・モジュールとの間の通信を制御する。メモリ・インターフェース・モジュール２４はまた、制御メッセージを処理する入出力ハードウェア２５を含む。制御記憶部２６は、ノードがそのデータ処理機能を実行できるようにマイクロプロセッサ２２を操作する制御コードを含む。電気的消去可能なプログラム可能読取り専用メモリ（ EEPROM）２８は、基本マイクロコードの記憶域を提供する。基本マイクロコードは、電源投入時またはリセット操作時にノードのブートストラップ始動を可能にする制御コードである。起動中に、基本マイクロコードは、メモリ・インターフェース機構２４を介して制御記憶部２６にロードされる。基本マイクロコードが制御記憶部２６内に現れると、マイクロプロセッサ２２が、システムのマイクロコードの残りの部分（たとえば、「機能」のマイクロコード）を、ディスク・ドライブ３２から装置インターフェース機構３４およびメモリ・インターフェース機構２４を介して制御記憶部２６にロードすることが可能になる。基本マイクロコードと機能マイクロコードが共にロードされたとき、制御記憶部２６は、ノードがすべてのデータ処理機能を実行できるようにするのに十分なマイクロコードを含む。

【００１２】複数のディスク・ドライブ３２（１台だけ示す）が、装置インターフェース機構３４を介してメモリ・インターフェース機構２４およびデータ・バッファ・インターフェース機構３５に接続される。データ・バッファ・インターフェース機構３５は、データ・バッファ３６をネットワーク・インターフェース機構３７に接続する。データ・バッファ３６は、（制御メッセージとは対照的に）入力と出力両方のデータ・メッセージにバッファ機能を提供する。さらにバッファ・インターフェース機構３５は、受け取ったデータの処理をする入出力ハードウェア・ポート３８を含む。バッファ・インターフェース機構３５内の入出力ハードウェア・ポート３８およびメモリ・インターフェース機構２４内の入出力ハードウェア・ポート２５は、制御記憶部２６内のエントリによって制御される。ネットワーク・インターフェース機構３７は、入力出力両方のメッセージ転送にインターフェース機能を提供する。

【００１３】マルチノード・システム１０（図1）の動作中、１つまたは複数のノードＡ～Ｆなど内の制御マイクロコードは、絶えず、更新、変更または修正される必要がある。本発明の特徴は、この機能が、マルチノード・システム１０を使用から外さずに実行されることである。マルチノード・システム１０の制御マイクロコードは、あるノードが一時的に使用から外されている場合でも、データ処理動作が継続できるようにする。そのために、あるノードが使用から外されている場合に動作する、１つまたは複数の他の重複したノードを設ける（たとえば、図1に破線で示したノードＦを参照）。別法として、更新されるノードに割り当てられていた仕事を、別の動作ノードに一時的に割り当て、あるいは、そのノードが更新された後、他のノードの更新中に実行されるように予定を立て直すこともできる。マイクロコードの変換を実施するために、各ノードにあるオペレーティング・ソフトウェアは、「作動中に」システムからノードを外し、リンクを仕切って、オンラインでネットワーク内のノードに戻すことができなければならない。これらの機能を実行する方法は本発明の範囲を越えるので、図１のノード・システムにそれらの機能があると想定すること以外はこれ以上詳しく説明しない。

【００１４】以下、コードが稼働状態にある間にコードに対する非破壊的な更新を可能にする、マイクロコードの「常時交換」手順を説明する。そのような「常時交換」動作を実施するために、マイクロコードは、２つのレベルのコードがシステム上で同時に実行される間に、２つのコードの間の一時的な非互換性に対処しなければならない。本明細書で利用するコードの常時交換処理は、変更がノード内部の動作だけに影響を及ぼすようなノード・レベル、また、変更が複数のノードの境界にまたがる場合、さらに、コードの変更が多数のノードに及ぶだけでなくノード間の更新の順序付けをも必要とする場合のいずれにおいてもコードの変更を導入する能力を提供するように構成される。

【００１５】最も簡単なタイプの常時交換コード変更は、すべてのコード修正が1つのノード内で行われ、別のノード内のコードの修正を必要とするようなノード・インターフェースに表れないものである。前述のように、システムは、あるノードの損失を許容し残りのノードで動作し続けるようにプログラムされているので、更新されるソフトウェア構成要素を含むノードはシステムの動作から外される。さらに、システムは、使用から外されたノードの一時的な代用として、待機ノード（たとえば、ノードＦ）を使用することができる。使用から外されたノードでは、改訂されたコードがそのノードにロードされ（全ノード・マイクロコードと想定して）、次にそのノードがリセットされ、オペレーティング・ソフトウェアが、更新済みコードと再びリンクされる。次い

9

で、ノードが動作に戻されて、そのノードの更新中に動作し続けていた他のノードと接合される。この手順は、一時にノード1つずつ、コードの更新が必要な他のノードがすべて終了するまで次々に行われる。

【００１６】先の説明では、全ノード・レベルのコードの更新を検討した。単一のノード内でモジュール・レベルの更新を実施するためには(すなわち、マイクロコード全体よりも少ない更新) 、改訂されたモジュールが導入され、その後、変更を必要とするモジュールが切り離され、ソフトウェアの構成要素の残りの部分が稼働し続ける間、その代替モジュールが、動的にコードにリンクされる。

【００１７】複数のノードが関係するコード改訂の常時交換を実施するために、本発明は、新しいコード単位用に書かれた他のコードを含み、新しいコード用に別のコード単位でインターフェース機能を操作する変換プログラム・コード・モジュールを使用する。変換プログラムは、ノード・レベルの更新、モジュール・レベルの更新、および影響を受けるノード間のインターフェースに至るあらゆるコードの更新に利用される。変換プログラムは、異なる技術変更レベルにある2つのコード単位間の一時的非互換性に対処する。改訂された各コード単位内で、影響を受ける1つのインターフェースのために1つの変換プログラムが設計され符号化される。改訂された同じコード単位が、別々のインターフェースを操作するために複数の変換プログラムを必要とすることがある。2つのコード単位間のインターフェースについては、両方のコード単位が変更されたとき、両者のコード・レベルを一致させるために、各コード単位に1つずつ、1対の変換プログラムが必要とされることがある。2つのコード単位の更新が連続している場合は、最初に更新されたコード単位だけが変換プログラムを必要とする。

【００１８】ノード・レベルの更新の際には、新しいレベルのコードで更新されたばかりのノードが、まだ従来の技術変更レベルにあるノードとインターフェースをとらなければならない。新しいレベルのコードは、それ自体の中に、影響を受けたインターフェースごとに1つの変換プログラムを含まなくてはならない。各変換プログラムは、別のノードで遭遇したとき、動作またはプロトコルを旧コードの同等の機能に送る働きをする。したがって、変換プログラムは、すべてのノードが同じ技術変更レベルに更新されるまで一時的ブリッジとして働く。

【００１９】他のモジュール、すなわち「シーケンサ」コード・モジュールは、マルチノード・システム内の多数のノードに対するコード変更の適用シーケンスを制御する。シーケンサは、影響を受けたコード単位を含むすべてのノードにすべての変更が導入されることを保証し、また、変更部分の間に順序依存関係がある場合は、導入が「台本(script) 」によって指定された順序になることを保証する。より具体的には、シーケンサの台本は、様々なコード単位およびノードの間の更新の順序付けを管理する。改訂コード用のシーケンサの台本が提供されない場合は、コード修正は、システム上に導入されたノードによる省略時の台本に従う。

10

【００２０】次に、図3を参照し、ノードAが、新しい動作マイクロコード５０を導入済みであり、ノードAとBならびにノードAとDの間でそれぞれ通信を可能にする変換プログラム５４と５６を含むと仮定する。さらに、シーケサンサ・モジュール５２が、別のノード(すなわち、ノードN) 内にあり、新しいコードの導入の順序を制御する台本を含むと仮定する。ノードBとDはさらに、古い動作マイクロコード５８と６０をそれぞれ含む。新しい動作マイクロコード５０がノードAにロードされると、ノードAが新しい動作マイクロコード５０の制御下で動作している場合でも、変換プログラム５４と５６が、ノードAがノードBとD両方と通信するように初期設定される。ノードBとDが更新されるまで、変換プログラム５４と５６は活動状態のまま留まり、ノードA、B、Dの間で連続した通信が可能になる。

【００２１】次に、図4に示したシーケンサ段階2が、ノードB内の旧動作マイクロコード５８を、変換プログラム・モジュール６４を含む新動作マイクロコード６２と取り替える。変換プログラム・モジュール６４は、新動作マイクロコード６２と、ノードDにまだ存在する旧動作マイクロコード６０との間の通信を可能にする。一方、今はノードB内に新動作マイクロコード６２が存在するので、ノードA内の新動作マイクロコード５０との直接の通信が可能になる。新動作マイクロコード６２がノードBに導入された後、変換プログラム５４は、切断されて、新動作マイクロコード５０と新動作マイクロコード６２の間の通信が可能になる。変換プログラム５４は、当分の間ノードA内にそのまま残る。ノードB内の変換プログラム６４は、新動作マイクロコード６２とノードD内の旧動作マイクロコード６０の間の通信を操作する。

【００２２】次に、シーケンサ段階3 (図5に示す) では、新動作マイクロコード６６をノードDに導入する。新動作マイクロコード６６の初期設定の後で、すべての変換プログラム上の通信が非活動化され、ノードA、BおよびDはそれぞれ、導入された新動作マイクロコードを有し、変換プログラム・モジュールを介さずに直接通信することが可能になる。

【００２３】次に、シーケンサ５２は、2つの追加の段階(４と５) に進みノードAとBから変換プログラム・モジュール５４、５６、および６４を除去させる(図6参照) 。この段階で、すべてのノードが新動作マイクロコードによって更新され、システムは完全に動作可能である。

【００２４】以上に示したように、ノードが動作したま

まで、新しいコード単位(および変換プログラム) が各ノードに順番にロードされる。したがって、ノードＡがロードされた最初のノードだと仮定すると、最初にノードＡが使用から外され、新しいコードがロードされる。ノードＡが使用に戻されると、ノードＡは、その変換プログラム・モジュールを介してシステム内の他のノードと通信し、それにより、システムが動作し続けることが可能になる。次に、ノードＢが使用から外されて新しいコード単位と変換プログラム・モジュールで更新され、使用に戻されたとき、システムは、ノードＡとＢ内の変換プログラム・モジュールによって動作し続け、コードの新バージョンと旧コード単位(たとえば、ノードＤにある) の間で通信を続けることが可能になる。

【００２５】新コード単位を含むノードから通信が要求されたときは、それに関連する変換プログラムが、通信が行われるべきノードにメッセージを送って、そのノード内にあるコードの技術変更(ＥＣ) レベルに関して照会する。図2 に示したように、コードのＥＣ変更レベルの指示は、メモリ・インターフェース機構２４中に維持される。ＥＣレベルの照会を受け取ったノードは、ＥＣレベルの指示によって応答する。受け取ったＥＣレベルが、照会したノードのものと同一である場合は、関連する変換プログラムは新しいインターフェースを利用する。ＥＣレベルが異なる場合は、未更新ノードと通信が行われるべきであることを示し、古いインターフェースが実施される。

【００２６】以上の説明は本発明の例証にすぎないことを理解されたい。当業者は、様々な代替例および修正例を、本発明から逸脱することなしに考案することができる。したがって、本発明は、添付の特許請求の範囲に含まれるそのような代替例、修正例および変形例をすべて含むものである。

【００２７】まとめとして、本発明の構成に関して以下の事項を開示する。

【００２８】(１) 多数のノードが通信ネットワークによって相互接続され、各ノードが、プロセッサ、メモリ、制御コード単位の第1 のバージョン、および前記制御コード単位に関する技術変更レベル(ＥＣＬ) 値を含み、１つのノードが除去されたときでも動作可能な、計算システムが動作したままで計算システムにおける複数のノード内の制御コードを更新する方法であって、

ａ．前記計算システムから第1 のノードを分離する段階と、

ｂ．前記制御コード単位の改訂バージョンと、前記第1 のノード内の前記プロセッサと協力して動作し前記第1 のノードと第2 のノードとの通信中に第1 と第2 のインターフェース機能を実行する変換プログラム・コード・モジュールとを、前記第1 のノードのメモリに導入して操作する段階と、

ｃ．前記第1 のノードを前記計算システムに結合する段階と、

ｄ．前記第1 のノードを操作して前記第2 のノードとの通信を必要とする機能を実行する段階とを含み、前記第1 のノード内の前記変換プログラム・コード・モジュールが前記第2 のノードに記憶されたＥＣＬ値を決定し、前記第1 と第2 のノード内のＥＣＬ値が一致する場合は、前記第1 のインターフェース機能を使って前記第2 のノードと通信し、前記第1 と第2 のノード内のＥＣＬ値が一致しない場合は、前記第2 のインターフェース機能を使って前記第2 のノードと通信し、それにより、前記制御コード単位の更新中に前記計算システムを動作可能にする方法。

(２) ｅ．前記計算システムから前記第2 のノードを分離する段階と、

ｆ．前記制御コード単位の改訂バージョンと、前記第2 のノード内の前記プロセッサと協力して動作し前記第2 のノード、第1 のノードおよびその他のノードの間の通信中に第1 と第2 のインターフェース機能を使用可能にし実行する変換プログラム・コード・モジュールとを、前記第2 のノードのメモリに導入する段階と、

ｇ．前記第2 のノードを前記計算システムに結合する段階と、

ｈ．前記第1 と第2 のノードを操作して、前記第1 のノード、第2 のノードおよびその他のノードの間の通信を必要とする機能を実行する段階とをさらに含み、前記変換プログラム・コード・モジュールが、前記第1 と第2 のノードに記憶されたＥＣＬ値の一致を確認したときには、前記第1 のインターフェース機能を使って前記第1 と第2 のノードの間の通信を可能し、前記その他のノードとのＥＣＬ値の不一致を確認したときは、前記第2 のインターフェース機能を使って前記その他のノードと通信し、それより、前記ノード全部の前記制御コード単位の更新中に前記第1 、第2 およびその他のノードを動作可能にする、上記(１) に記載の方法。

(３) 計算システムが、前記計算システムのノードに修正された前記制御コード単位を導入する順序を定義するシーケンサ・コード・モジュールを含み、前記制御コード単位の前記改訂バージョンを、前記シーケンサ・コード・モジュールによって指定されたように前記ノードに導入する段階を含むことを特徴とする、上記(１) に記載の方法。

(４) あるノードが、前記制御コード単位の第1 バージョンおよび改訂バージョンを有する他のノードとの相互通信を必要とする場合、そのノードに導入された変換プログラム・コード・モジュールが、インバウンドおよびアウトバウンドのコード変換プログラム・サブモジュールを含むことを特徴とする、上記(２) に記載の方法。

(５) 複数のノードが通信ネットワークによって相互接続され、各ノードが、プロセッサ、メモリ、制御コード単位の第1 のバージョン、および前記制御コード単位に

13

関する技術変更レベル(ＥＣＬ) 値を含み、１つのノードが除去されたときでも動作可能な、計算システムが動作したままで計算システムにおける複数のノード内の制御コードを更新する装置であって、前記第1 のノードを前記計算システムから切り離した後、前記制御コード単位の改訂バージョンと、前記第1 のノード内の前記プロセッサと協力して動作し前記第1 のノードと第2 のノードとの通信中に第1 と第2 のインターフェース機能を実行する変換プログラム・コード・モジュールとを、前記第1 のノードのメモリに導入する手段と、前記第1 のノードを前記計算システムに再結合し、前記第1 のノードを操作して、前記第2 のノードとの通信を必要とする機能を実行する手段と、前記第1 のノード内の前記変換プログラム・コード・モジュールを操作して、前記第2 のノードに記憶されたＥＣＬ値を決定し、前記第1 と第2 のノード内のＥＣＬ値が一致する場合は、前記第1 のインターフェース機能を使って前記第2 のノードと通信し、前記第1 と第2 のノード内のＥＣＬ値が一致しない場合は、前記第2 のインターフェース機能を使って前記第2 のノードと通信する手段とを含む装置。

(６) 前記計算システムから前記第2 のノードを切り離す際に、前記制御コード単位の改訂バージョンと、前記第2 のノード内の前記プロセッサと協力して動作し前記第2 のノード、第1 のノードおよびその他のノードの間の通信中に第1 と第2 のインターフェース機能を使用可能にして実行する変換プログラム・コード・モジュールとを、前記第2 のノードのメモリに導入する手段と、前記第2 のノードを前記計算システムに再結合した後で動作可能な、前記第1 と第2 のノードを操作して、前記第1 のノード、第2 のノードおよびその他のノードの間の通信を必要とする機能を実行する手段と、前記第1 と第2 のノードに一致するＥＣＬ値が記憶されているかどうか決定し、一致するＥＣＬ値が記憶されている場合は、前記制御コード・モジュール内の前記第1 のインターフェース機能を使って前記第1 と第2 のノードの間の通信を可能し、前記第2 のノードおよび前記その他のノード内に不一致のＥＣＬ値が確認された場合は、前記制御コード・モジュール内の前記第2 のインターフェース機能を使って前記その他のノードと通信する手段とをさらに含むことを特徴とする、上記(５) に記載の装置。

(７) 前記装置が、修正された前記制御コード単位を前記計算システムのノード内に導入する順序を定義するシーケンサ・コード・モジュールを含み、各ノードにおいて、前記制御コード単位の前記改訂バージョンを、前記シーケンサ・コード・モジュールによって指定された順序で導入するために前記シーケンサ・コード・モジュールと協力して動作する手段を含むことを特徴とする、上記(６) に記載の装置。

(８) あるノードが、前記制御コード単位の第1 バージョンおよび改訂バージョンを有する他のノードとの相互

14

通信を必要とする場合、そのノードに導入された変換プログラム・コード・モジュールが、インバウンドおよびアウトバウンドのコード変換プログラム・サブモジュールを含むことを特徴とする上記(７) に記載の装置。

(９) 複数のノードが通信ネットワークによって相互接続され、各ノードが、プロセッサ、メモリ、前記ノード内の動作マイクロコードのコード・モジュールである制御コード単位の第1 のバージョン、および前記制御コード単位に関する技術変更レベル(ＥＣＬ) 値を含み、計算システムが動作したままで計算システムにおける多数のノード内の制御コードを更新する方法であって、

ａ．第1 のノード内の前記動作マイクロコードから制御コード単位を分離する段階と、

ｂ．前記分離した制御コード単位の代わりに、前記制御コード単位の改訂バージョンと、前記第1 のノード内の前記プロセッサと協力して動作し前記第1 のノードと第2 のノードとの通信中に第1 と第2 のインターフェース機能を実行する変換プログラム・コード・モジュールとを、前記第1 のノード内のメモリに導入し操作する段階と、

ｃ．前記第1 のノードを操作して前記第2 のノードとの通信を必要とする機能を実行する段階とを含み、前記第1 のノード内の前記変換プログラム・コード・モジュールが前記第2 のノードに記憶されたＥＣＬ値を決定し、前記第1 と第2 のノード内のＥＣＬ値が一致する場合は、前記第1 のインターフェース機能を使って前記第2 のノードと通信し、前記第1 と第2 のノード内のＥＣＬ値が一致しない場合は、前記第2 のインターフェース機能を使って前記第2 のノードと通信し、それにより、前記制御コード単位の更新中に前記計算システムを動作可能にする方法。

(１０) ｄ．前記第2 のノード内の前記マイクロコードから前記制御コードを分離する段階と、

ｅ．前記分離された制御コード単位の代わりに、前記制御コード単位の改訂バージョンと、前記第2 のノード内の前記プロセッサと協力して動作し前記第2 のノード、第1 のノードおよびその他のノードの間の通信中に第1 と第2 のインターフェース機能を使用可能にし実行する変換プログラム・コード・モジュールとを、前記第2 のノード内のメモリに導入する段階と、

ｆ．前記修正された前記制御コード単位を前記第2 のノード内の動作マイクロコードに結合する段階と、

ｇ．前記第1 と第2 のノードを操作して、前記第1 のノード、第2 のノードおよびその他のノードの間での通信を必要とする機能を実行する段階とをさらに含み、前記変換プログラム・コード・モジュールが、前記第1 と第2 のノードに記憶されたＥＣＬ値の一致を確認したときには、前記第1 のインターフェース機能を使って前記第1 と第2 のノードの間の通信を可能し、前記その他のノードとのＥＣＬ値の不一致を確認したときは、前記第2

15

のインターフェース機能を使って前記その他のノードと通信し、それより、前記第1 、第2 およびその他のノードを、前記ノード全部の前記制御コード単位の更新中に動作可能にする、上記(９) に記載の方法。

(１１) 計算システムが、前記計算システムのノードに前記制御コード単位の改訂バージョンを導入する順序を定義するシーケンサ・コード・モジュールを含み、前記制御コード単位の改訂バージョンを、前記シーケンサ・コード・モジュールによって指定されるように前記ノードに導入する段階を含むことを特徴とする、上記(９) に記載の方法。

(１２) 前記ノードが、前記制御コード単位の第1 バージョンおよび改訂バージョンを有する他のノードとの相互通信を必要とする場合、ノード内に導入された変換プログラム・コード・モジュールが、インバウンドおよびアウトバウンドのコード変換プログラム・サブモジュールを含むことを特徴とする、上記(９) に記載の方法。

【００２９】

【発明の効果】本発明の実施により、コード変更が異なるレベルにあるノード間での通信を可能にするコード更新がマルチノード・システムにおいて可能になる。

【００３０】また、導入処理の間もマルチノード・システムが動作し続けるように制御コードの更新を導入する、マルチノード・システム用の装置を提供することができる。

【００３１】また、コードの改訂が所定の順序に従って導入される、マルチノード・システムにコードの改訂を導入するための方法および装置が提供できる。

【図面の簡単な説明】

【図1 】ディスク・ドライブのメモリ能力を接続されたホスト・プロセッサに提供するための、多数のノードを含むマルチノード・システムを示すブロック図である。

【図2 】図1 のシステムに利用される典型的なノードのブロック図である。

【図3 】異なる変更レベルの導入済み制御コードを有する様々なノードで本発明を実施する方法を示すブロック図である。

【図4 】異なる変更レベルの導入済み制御コードを有する様々なノードが本発明を実施する方法を示すブロック図である。

【図5 】異なる変更レベルの導入済み制御コードを有する様々なノードが本発明を実施する方法を示すブロック図である。

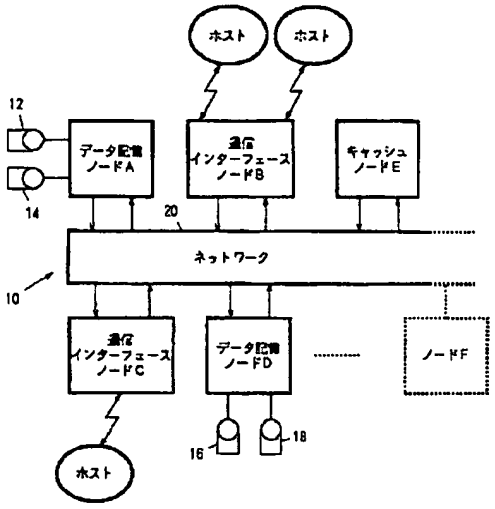
【図6 】異なる変更レベルの導入済み制御コードを有する様々なノードが本発明を実施する方法を示すブロック図である。

【符号の説明】

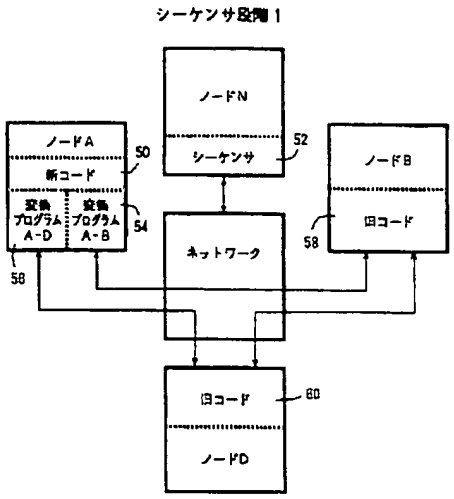
１０　ディスク・ドライブ・アレイ
１２　ディスク・ドライブ
１４　ディスク・ドライブ
１６　ディスク・ドライブ
１８　ディスク・ドライブ
２０　通信ネットワーク
２２　マイクロプロセッサ
２４　メモリ・インターフェース・モジュール
２６　制御記憶部
２８　プログラム可能読取り専用メモリ
３２　ディスク・ドライブ
３４　装置インターフェース機構
３５　データ・バッファ・インターフェース機構
３６　データ・バッファ
３７　ネットワーク・インターフェース機構
３８　入出力ハードウェア・ポート
５０　新動作マイクロコード
５２　シーケンサ・モジュール
５４　変換プログラム
５６　変換プログラム
５８　旧動作マイクロコード
６０　旧動作マイクロコード

16

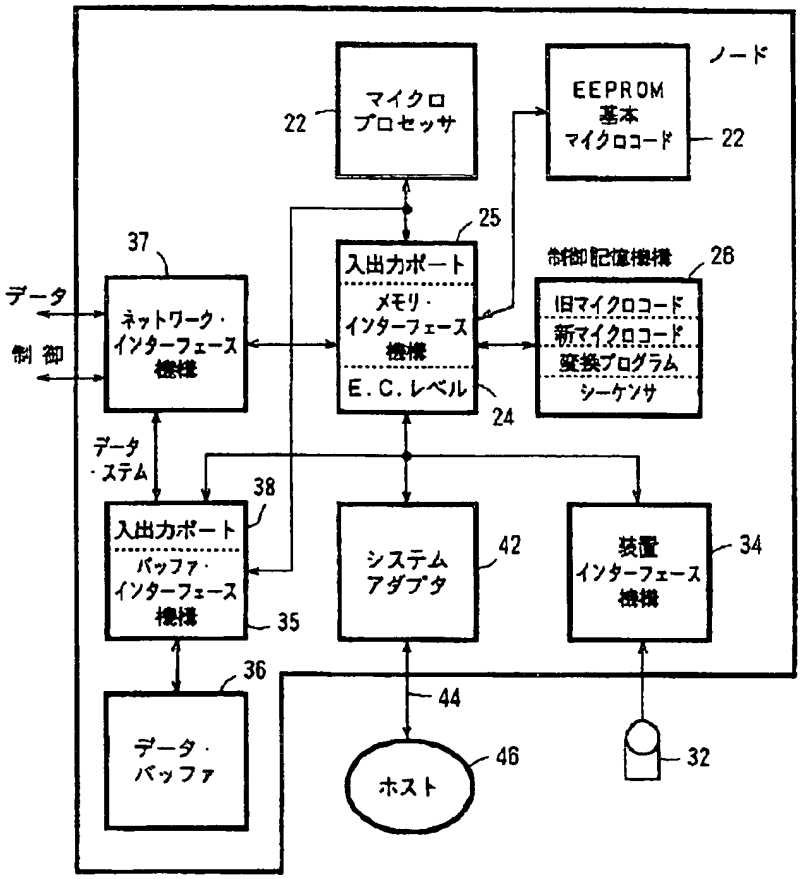
【図1】



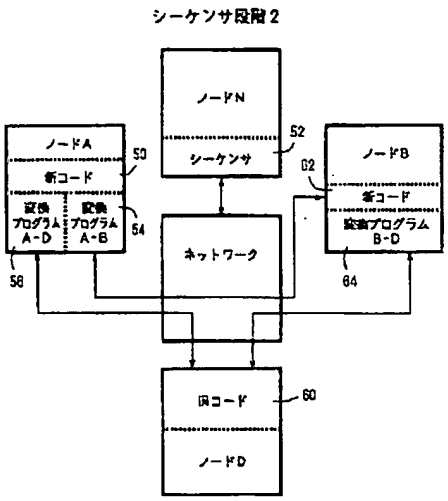
【図3】



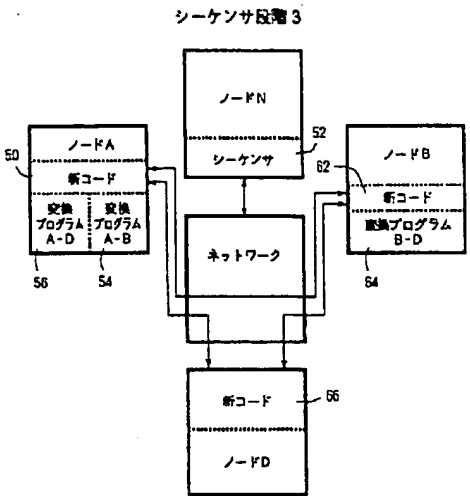
【図2】



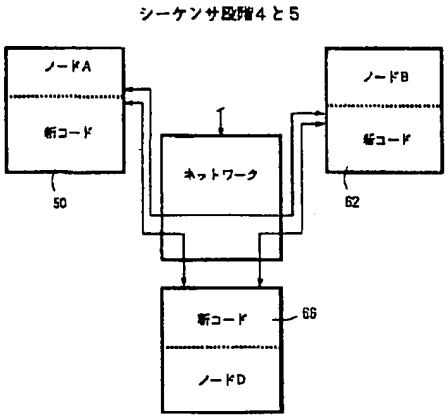
【 図4 】



【 図5 】



【 図6 】



フロント ページの続き

(72)発明者　ローレンス・ワイ・ホー
アメリカ合衆国95037　カリフォルニア州　モーガン・ヒル　オーク・リーフ・ドライブ　16945

(72)発明者　チェスター・アール・スティーヴンス
アメリカ合衆国95123　カリフォルニア州　サンノゼ　アズール・アベニュー　3602

(72)発明者　ジェームズ・ティー・ブレイディー
アメリカ合衆国95120　カリフォルニア州　サンノゼ　クイーンズフリッジ・コート　1060

(72)発明者　デーヴィッド・ティー・ワン
アメリカ合衆国95120　カリフォルニア州　サンノゼ　アンジュー・クリーク・サークル　7023